# Capturing Missing Tuples and Missing Values

# Capturing Missing Tuples and Missing Values

Wenfei Fan
University of Edinburgh &
Harbin Institute of Technology
wenfei@inf.ed.ac.uk

Floris Geerts
School of Informatics
University of Edinburgh
fgeerts@inf.ed.ac.uk

## Abstract

Databases in real life are often neither entirely closed-world nor entirely open-world. Indeed, databases in an enterprise are typically *partially closed*, in which a part of the data is constrained by master data that contains complete information about the enterprise in certain aspects [21]. It has been shown that despite missing tuples, such a database may turn out to have complete information for answering a query [9].

This paper studies partially closed databases from which *both tuples and values* may be missing. We specify such a database in terms of conditional tables constrained by master data, referred to as $c$-instances. We first propose three models to characterize whether a $c$-instance $\mathcal{T}$ is *complete* for a query $Q$ *relative to* master data. That is, depending on how missing values in $\mathcal{T}$ are instantiated, the answer to $Q$ in $\mathcal{T}$ remains unchanged when new tuples are added. We then investigate four problems, to determine (a) whether a given $c$-instance is complete for a query $Q$, (b) whether there exists a $c$-instance that is complete for $Q$ relative to master data available, (c) whether a $c$-instance is a minimal-size database that is complete for $Q$, and (d) whether there exists a $c$-instance of a bounded size that is complete for $Q$. We establish matching lower and upper bounds on these problems for queries expressed in a variety of languages, in each of the three models for specifying relative completeness.

**Categories and Subject Descriptors:** H.2.3 [**Information Systems**]: Database Management – *Languages*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic — *Computational Logic*
**General Terms:** Languages, Theory, Design.

## 1. Introduction

Incomplete information has been a longstanding issue. The scale of the problem is such that it is common to find critical information missing from databases. For instance, it is estimated that pieces of information perceived as being needed for clinical decisions were missing from 13.6% to 81% of the time [23]. Traditional work on this issue adopts either the Closed World Assumption (CWA) or the Open World Assumption (OWA). The CWA assumes that a database has collected all the tuples representing real-world entities, but the *values* of some attributes in those tuples are possibly *missing*. The OWA assumes that some *tuples* that represent real-world entities may also be *missing* (see [2, 30] for surveys).

| | name | str | city | zip | YoB | cond |
|---|---|---|---|---|---|---|
| $t_1$: | John | 3 Elm | Edi | EH8 9AB | 2000 | |
| $t_2$: | $x$ | $y$ | Edi | EH8 9AB | $z$ | ($x \neq$ John) |
| $t_3$: | Mary | 5 Mayfield | $w$ | $u$ | 2000 | ($w \neq$ Edi) |

**Figure 1: A $c$-table of Patient.**

Real-life databases are, however, often neither entirely closed-world nor entirely open-world. This is particularly evident in Master Data Management (MDM), one of the fastest growing software markets [22, 27]. Master data is a single repository of high-quality data that provides various applications with a synchronized, consistent view of the core business entities of an enterprise [21]. It is a closed-world database about the enterprise in certain aspects, *e.g.,* employees and customers. In the presence of master data, databases of the enterprise are typically *partially closed* [9]. While parts of their data are constrained by the master data, *e.g.,* employees and customers, the other parts of the databases are open-world, *e.g.,* sale transactions and service records.

Partially closed databases have recently been studied in [9], in the absence of missing values. Certain information in a partially closed database $\mathcal{I}$ is bounded by master data $D_m$, specified by a set $V$ of containment constraints (CCs) from $\mathcal{I}$ to $D_m$. *Relative to* $D_m$, $\mathcal{I}$ is said to be *complete* for a query $Q$ if $Q(\mathcal{I}) = Q(\mathcal{I}')$ for every partially closed extension $\mathcal{I}'$ of $\mathcal{I}$, *i.e.,* $\mathcal{I}' \supset \mathcal{I}$ such that $(\mathcal{I}', D_m)$ satisfies $V$. That is, adding new tuples to $\mathcal{I}$ either does not change the query answer or violates the CCs. It is shown in [9] that despite missing tuples, a partially closed database may still have complete information for answering queries.

The work of [9] has focused on *ground instances*, namely, database instances from which tuples are possibly missing, but all the values of the existing tuples are in place. In practice, however, *both tuples* and *values* are commonly found *missing* from a database. This introduces new challenges to characterizing and determining whether a database is complete for a query relative to master data.

**Example 1.1:** Consider a database $D$ of UK patients, specified by schema Patient(name, str, city, zip, YoB). Consider a query $Q_1$ to find the streets of those patients who live in Edi with zip = 'EH8 9AB' and were born in 2000. One can hardly trust the answer $Q_1(D)$ since tuples may be missing from $D$, even when no values of the tuples in $D$ are missing.

Not all is lost. Indeed, suppose that master data $D_m$ is

available, specified by schema $\mathsf{Patient}_m(\mathsf{name}, \mathsf{str}, \mathsf{zip}, \mathsf{YoB})$, which maintains a complete record of those patients living in Edi and born after 1990. Then we can conclude that $Q_1$ finds a complete answer in $D$ if $Q_1(D)$ returns the streets of all patients $p$ in $D_m$ with $p[\mathsf{zip}]$='EH8 9AB' and $p[\mathsf{YoB}]$=2000. Indeed, there is not need to add new tuples to $D$ in order to answer $Q_1$. Relative to master data $D_m$, the seemingly incomplete database $D$ turns out to be complete for $Q_1$.

In practice, values may also be missing. Following [13, 14], we use a conditional table ($c$-table) $T$ to represent such a database, as shown in Fig. 1. In "tuple" $t_2$ of $T$, the values of $t_2[\mathsf{name}]$, $t_2[\mathsf{str}]$ and $t_2[\mathsf{YoB}]$ are missing, and the condition $t_2[\mathsf{cond}]$ tells us that $t_2[\mathsf{name}]$ is not John; similarly for $t_3$.

To characterize whether $T$ is complete for $Q_1$, we have to decide how to fill in the missing values in $T$.

(1) One may want $T$ to be *strongly complete* for $Q_1$, *i.e.,* for each valuation $\mu$ of the variables in $T$, $\mu(T)$ is complete for $Q_1$ relative to $D_m$. In other words, $T$ has complete information for answering $Q_1$ despite missing tuples and values.

(2) One may also want $T$ to be *weakly complete*, *i.e.,* all *the certain answers* to the query $Q_1$ can already be found in $T$.

(3) Alternatively, one may want $T$ to be *completable*, when there exists a valuation $\mu$ of $T$ such that $\mu(T)$ is complete for $Q_1$ relative to $D_m$. That is, when the missing values are correctly instantiated, $T$ has complete information to answer query $Q_1$. □

These suggest that relatively complete databases have to accommodate not only missing tuples but also missing values. In addition, there are several fundamental questions that are not only of theoretical interest, but are also important to database users and developers. For instance, a user may naturally ask whether a database in use is complete for a query relative to master data. A developer may want to know what is a minimal amount of information one has collect to build a relatively complete database, and moreover, whether the database has a bounded size. These practical needs call for a full treatment of relative information completeness.

**Relative information completeness**. To capture missing values and missing tuples, we extend the notion of partially closed databases [9] to $c$-instances. A $c$-instance is a collection of $c$-tables [13, 14] in which certain parts are bounded by master data, via a set of containment constraints (CCs).

*Models*. We propose three models to specify whether a $c$-instance $\mathcal{T}$ is complete for a query $Q$ relative to master data $D_m$. As illustrated in Example 1.1, $\mathcal{T}$ is (1) *strongly complete* if each valuation of $\mathcal{T}$ yields a ground instance that is complete for $Q$ relative to $D_m$; (2) *weakly complete* if one can find in $\mathcal{T}$ the certain answers to $Q$ over all partially closed extensions of valuations of $\mathcal{T}$; and (3) *completable* if there exists a valuation of $\mathcal{T}$ that leads to a relatively complete database for $Q$. A user may choose a model that best serves her need.

*Data consistency*. We are interested in databases that are both relatively complete and consistent. The consistency of data is typically specified by integrity constraints, such that errors and conflicts in the data can be detected as violations of the constraints [4, 6]. We investigate the impact of integrity constraints on the analysis of relative completeness. In addition, instead of using a separate language of integrity constraints, we adopt a class of CCs that is also capable of expressing constraints commonly used in data cleaning.

*Analysis of c-instances*. We provide complexity bounds on basic issues in connection with $c$-instances. These problems are to decide, given a $c$-instance $\mathcal{T}$, (a) whether $\mathcal{T}$ makes sense, *i.e.,* whether there is any partially closed database represented by $\mathcal{T}$, and (b) whether $\mathcal{T}$ is extensible, *i.e.,* whether there exists any partially closed extension of $\mathcal{T}$.

**Main complexity results**. We identify four fundamental problems associated with relative information completeness, denoted by RCDP, RCQP, MinP and BdnP. Given a query $Q$ and master data $D_m$, (a) RCDP is to decide whether a database is complete for $Q$ relative to $D_m$, (b) RCQP asks whether it is possible to build a database complete for $Q$ relative to $D_m$, (c) MinP is to determine whether a database has a minimal size among those complete for $Q$ relative to $D_m$, and (d) BdnP asks whether there exists a database of a bounded size that is complete for $Q$ relative to $D_m$.

We investigate these problems *w.r.t.* several dichotomies:

- $\mathcal{L}_Q$: the query language in which $Q$ is expressed, ranging over conjunctive queries, (CQ), union of conjunctive queries (UCQ), positive existential FO queries ($\exists$FO$^+$), first-order queries (FO), and FP, all with inequality ($\neq$);
- $c$-instances vs. ground instances, *i.e.,* in the presence or in the absence of missing values; and
- different models of relative completeness, *i.e.,* when a $c$-instance is required to be strongly complete, weakly complete or completable for $Q$, relative to $D_m$.

We provide a comprehensive picture of these problems with different combinations of these factors. We establish their lower and upper bounds, *all matching*, ranging over $O(1)$, coDP, $\Pi_2^p$, $\Delta_3^p$, $\Sigma_3^p$, $\Pi_3^p$, $\Sigma_4^p$, NEXPTIME, coNEXPTIME, and undecidable.

Our main conclusions are as follows.

(a) These problems are decidable for CQ, UCQ and $\exists$FO$^+$, but are mostly undecidable for FO and FP. However, they are decidable for FP in the weak completeness model. Some problems for CQ and UCQ behave differently.

(b) The presence of missing values makes our lives harder when RCDP and MinP are concerned. For example, MinP for CQ is $\Delta_3^p$-complete for ground instances in the strong model, while it is $\Pi_3^p$-complete for $c$-instances. In contrast, it does not complicate the analyses of RCQP and BdnP.

(c) These problems have rather diverse complexity in different models of relative completeness. For instance, RCQP for FP is undecidable in the strong model, but is trivially decidable for weakly complete $c$-instances. On the other hand, MinP for UCQ is $\Pi_3^p$-complete for strongly complete $c$-instances but it becomes $\Sigma_4^p$-complete in the weak model.

To our knowledge, this work is a first treatment of relatively complete databases in the presence of both missing values and missing tuples. We identify important problems associated with partially closed $c$-instances, and provide matching complexity bounds on these problems. A variety of techniques are used to prove these results, including finite-model theoretic constructions, characterizations of relatively complete databases and a wide range of reductions.

**Related work**. This work extends [9] by dealing with missing values and providing a variety of complexity bounds for

new decision problems. We propose three models for relatively complete $c$-instances, which were not considered in [9]. For ground instances in the strong model, RCDP and RCQP have been studied in [9], with several cases left open there. However, none of RCDP, RCQP, MinP and BdnP has been studied either for $c$-instances or for weakly complete databases (ground or not). Furthermore, no previous work has studied MinP and BdnP, even for ground instances.

There has been a host of work on incomplete information, notably representation systems (see [2, 30] for surveys, and more recently, [25]). This work adopts $c$-tables [13, 14] to represent databases with missing values. Our weak model for relative completeness is based on the certain answer semantics [14], and the strong model has a resemblance to strong representation systems. In contrast, completable $c$-instances do not find a counterpart in [13, 14]. The basic issues for $c$-instances (see Section 3) are similar to the problems studied in [3], but with master data. As opposed to prior work in this area, we aim to model partially closed databases commonly found in MDM, and to settle their associated decision problems that have not been studied before.

Several approaches have been proposed to modeling databases with missing tuples (*e.g.,* [12, 18, 24, 31]). A notion of *open null* was introduced in [12] to model locally controlled open-world databases, in which tuples or values can be marked with open null, while the rest of the data is closed-world. Complete and consistent extensions of an incomplete database were studied in [31]. There has also been work on modeling negative information via logic programming (see [30]). Neither master data nor the decision problems studied in this work have been considered there.

Closer to this work are partially complete databases studied in [18, 24], which assume a virtual database $D_c$ that contains complete information in *all relevant aspects*, and assume that any database $D$ either *contains* or is defined as *views* of $D_c$. A notion of answer completeness was proposed there, for deciding whether a query posed on $D_c$ can be answered in $D$. We assume neither the existence of $D_c$ with entire complete information nor views that define $D$ in terms of $D_c$. Furthermore, neither missing values nor the problems studied here were considered in [18, 24].

Certain answers have also been studied in data integration and data exchange. In data integration, for a query $Q$ posed on a global database $D_G$, one wants to find the certain answers to $Q$ over all data sources that are consistent with $D_G$ *w.r.t.* view definitions (see *e.g.,* [1, 17]). In data exchange, one wants to find the certain answers to a query over all target databases transformed from data sources via schema mapping (see [15, 5]). The decision problems studied here are not considered in data exchange or data integration. There has also been work on answering queries using views to decide, *e.g.,* whether views determine queries [28]. Our decision problems cannot be reduced to the problems studied there, and vice versa, because in MDM, one often cannot characterize databases as views of master data.

The study of query equivalence under constraints is quite different from this work. Indeed, the former is to determine the equivalence of different queries on all instances. In contrast, relative information completeness requires that the answer to the same query remains unchanged over partially closed extensions and possible valuations of missing values.

There has also been work on consistent query answering (*e.g.,* [4, 6]), to find certain answers to a query over all repairs of a database. Master data is not considered there, and we do not consider database repairs in this work.

Except for [9] as remarked above, we are not aware of any previous work on RCQP, MinP or BdnP. For ground instances in the strong model, RCDP is similar to the problem of query independence from updates [7, 20]. A revision of RCDP was recently studied in [11] for data exchange. None of the results of [7, 20, 11] carries over to our setting.

**Organization**. Section 2 presents three models for specifying relatively complete $c$-instances. Section 3 investigates the impact of integrity constraints and basic issues in connection with $c$-instances. Problems RCDP, RCQP, MinP and BdnP are studied in Sections 4, 5 and 6 for strongly complete, weakly complete and completable $c$-instances, respectively. Section 7 summarizes the main results and identifies open problems.

## 2. Relative Information Completeness Revisited

In this section we first review relatively complete ground instances [9]. We then present three models to characterize relatively complete $c$-instances. Finally we state the decision problems associated with relative information completeness.

### 2.1 Relatively Complete Ground Instances

A relational schema $\mathcal{R}$ is a collection $(R_1, \ldots, R_n)$ of relation schemas. Each $R_i$ is defined over a set of attributes. This set of attributes is also denoted by $R_i$. For each attribute $A$ in $R_i$, its (finite or infinite) domain is a set of constants, denoted as $dom(A)$.

**Ground instances and master data**. A *ground instance* $\mathcal{I}$ of $\mathcal{R}$ is of the form $(I_1, \ldots, I_n)$, where for each $i \in [1, n]$, $I_i$ is an instance of $R_i$ *without missing values*. That is, for each $t \in I_i$ and each $A \in R_i$, $t[A]$ is a constant in $dom(A)$.

Master data $D_m$ is a ground instance of a relational schema $\mathcal{R}_m$. It is a consistent and closed-world database.

**Partially closed databases**. We specify the relationship between a database and master data in terms of *containment constraints* (CCs). A CC $\psi$ is of the form $q(\mathcal{R}) \subseteq p(\mathcal{R}_m)$, where $q$ is a conjunctive query (CQ) defined over schema $\mathcal{R}$, and $p$ is a projection query over schema $\mathcal{R}_m$.

A ground instance $\mathcal{I}$ of $\mathcal{R}$ and master data $D_m$ of $\mathcal{R}_m$ *satisfy* $\psi$, denoted by $(\mathcal{I}, D_m) \models \psi$, if $q(\mathcal{I}) \subseteq p(D_m)$.

Intuitively, CWA is asserted for $D_m$, which imposes an upper bound on the information extracted by $q(\mathcal{I})$ from the database $\mathcal{I}$. On the other hand, OWA is assumed on the part of $\mathcal{I}$ that is not constrained by CCs.

**Example 2.1:** Recall the database $D$ and master data $D_m$ described in Example 1.1. We specify a set $V$ of CCs such that for each $y$ in [1991, 2009], $V$ includes $q_y(\mathsf{Patient}) \subseteq D_m$, where $q_y(n, s, z, d)$ is $\mathsf{Patient}(n, s, c, z, y) \wedge c =$ 'Edi'. These CCs assure that $D_m$ is an upper bound on the information in $D$ about patients living in Edi and born after 1990.

As will be seen in Section 3, certain integrity constraints can also be expressed as CCs. For example, consider a functional dependency (FD) $\phi$: $(\mathsf{zip} \rightarrow \mathsf{city}, \mathsf{str})$, *i.e.,* in the UK, zip code determines the city and street. Assume that master data contains an empty relation $D_\emptyset$. Then $\phi$ can be written as two CCs included in $V$: $q_{\mathsf{city}} \subseteq D_\emptyset$ and $q_{\mathsf{str}} \subseteq D_\emptyset$, where

$$q_{\mathsf{city}} = \exists z\ n_1 n_2\ s_1 s_2\ c_1 c_2\ d_1 d_2(\mathsf{Patient}(n_1, s_1, c_1, z, d_1) \wedge \mathsf{Patient}(n_2, s_2, c_2, z, d_2) \wedge c_1 \neq c_2),$$

which detects violations of $\mathsf{zip} \to \mathsf{city}$; similarly for $q_{\mathsf{str}}$. Note that we allow inequalities in CQ and hence, in CCs. □

We say that $(\mathcal{I}, D_m)$ satisfies a set $V$ of CCs, denoted by $(\mathcal{I}, D_m) \models V$, if $(\mathcal{I}, D_m) \models \psi$ for each $\psi \in V$.

A ground instance $\mathcal{I}$ of $\mathcal{R}$ is said to be *partially closed w.r.t.* $(D_m, V)$ if $(\mathcal{I}, D_m) \models V$. That is, the information in $\mathcal{I}$ is partially bounded by $D_m$ via the CCs in $V$.

**Relatively complete ground instances**. Consider ground instances $\mathcal{I} = (I_1, \ldots, I_n)$ and $\mathcal{I}' = (I'_1, \ldots, I'_n)$ of $\mathcal{R}$. We say that $\mathcal{I}'$ *extends* $\mathcal{I}$, denoted by $\mathcal{I} \subset \mathcal{I}'$, if for all $i \in [1, n]$, $I_i \subseteq I'_i$, and there is $j \in [1, n]$ such that $I_j \subset I'_j$.

The set of *partially closed extensions* of $\mathcal{I}$ is defined as:

$$\mathsf{Ext}(\mathcal{I}, D_m, V) = \{\mathcal{I}' \mid \mathcal{I} \subset \mathcal{I}', (\mathcal{I}', D_m) \models V\},$$

*i.e.,* for each $\mathcal{I}'$ in the set, (a) $\mathcal{I}'$ expands $\mathcal{I}$ by including new tuples, and (b) $\mathcal{I}'$ is partially closed *w.r.t.* $(D_m, V)$. We write $\mathsf{Ext}(\mathcal{I}, D_m, V)$ as $\mathsf{Ext}(\mathcal{I})$ when $D_m$ and $V$ are clear from the context.

A ground instance $\mathcal{I}$ is said to be *complete for a query $Q$ relative to* $(D_m, V)$ if for each $\mathcal{I}' \in \mathsf{Ext}(\mathcal{I})$, $Q(\mathcal{I}) = Q(\mathcal{I}')$.

That is, the answer to $Q$ in $\mathcal{I}$ remains unchanged no matter what new tuples are added to $\mathcal{I}$. Intuitively, $\mathcal{I}$ already has complete information for answering $Q$. The completeness is *relative to* $(D_m, V)$: the extensions must satisfy $V$.

**Example 2.2:** Recall $D, D_m$ and $Q_1$ from Example 1.1, and $V$ from Example 2.1. Then as shown in Example 1.1, $D$ is complete for $Q_1$ relative to $(D_m, V)$.

Consider a query $Q_2$ to find the streets of all patients born after 2000 and having zip code EH1 3CD. Suppose that there are such patient records in $D_m$, but $Q_2(D)$ is empty. Then $D$ is not complete for $Q_2$. However, we can make $D$ complete for $Q_2$ by adding to $D$ a *single* tuple $t$ with $t[\mathsf{zip}]$ = 'EH1 3CD'. Indeed, $V$ includes the CCs coding the FD $\phi$, assuring that there exists at most one street with this zip. Thus the expanded $D$ is complete for $Q_2$ relative to $(D_m, V)$.

In contrast, consider $Q_3$ to find the names of all patients born in 2000. Then $D_m$ does not help: it has no information about patients living in cities other than Edi. In this case we cannot make $D$ complete for $Q_3$ relative to $(D_m, V)$. □

### 2.2 Accommodating Missing Values

To specify databases with missing values, we adopt *conditional tables* (*c*-tables) [13, 14] with variables and local conditions. To define *c*-tables, for each relation schema $R_i$ and each attribute $A$ in $R_i$, we assume a countably infinite set $\mathsf{var}(A)$ of *variables* such that $\mathsf{var}(A) \cap dom(A) = \emptyset$, and $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$ for any attribute $B$ distinct from $A$.

**Partially closed *c*-instances**. A *c-table* of $R_i$ is a pair $(T, \xi)$, where (a) $T$ is a tableau in which for each tuple $t$ and each attribute $A$ in $R_i$, $t[A]$ is a constant in $dom(A)$ or a variable in $\mathsf{var}(A)$; and (b) $\xi$ associates a condition $\xi(t)$ with each tuple $t$ in $T$. The condition $\xi(t)$ is built up from atoms $x = y$, $x \neq y$, $x = c$, $x \neq c$, by closing under conjunction $\wedge$, where $x, y$ are variables and $c$ is a constant.

For example, a *c*-table is shown in Fig. 1.

A *valuation $\mu$ of* $(T, \xi)$ is a mapping such that for each tuple $t$ in $T$ and each attribute $A$ in $R$, $\mu(t[A])$ is a constant in $dom(A)$ if $t[A]$ is a variable, and $\mu(t[A]) = t[A]$ if $t[A]$ is a constant. Let $\mu(t)$ be the tuple of $R$ obtained by substituting $\mu(x)$ for each occurrence of $x$ in $t$. Then we define

$$\mu(T) = \{\mu(t) \mid t \in T, \mu \text{ satisfies } \xi(t)\},$$

*i.e.,* $\mu(t)$ is included in $\mu(T)$ iff $\xi(\mu(t))$ evaluates $\mathsf{true}$. Here $\mu(T)$ is a ground instance, *without* variables or conditions. That is, $(T, \xi)$ represents a set of possible worlds $\mu(T)$ when $\mu$ ranges over all valuations of $(T, \xi)$. We write $(T, \xi)$ simply as $T$ when $\xi$ is clear from the context.

A *c-instance* $\mathcal{T}$ of $\mathcal{R}$ is of the form $(T_1, \ldots, T_n)$, where for each $i \in [1, n]$, $T_i$ is a *c*-table of $R_i$. A *valuation $\mu$ of* $\mathcal{T}$ is $(\mu_1, \ldots, \mu_n)$, where $\mu_i$ is a valuation of $T_i$. We use $\mu(\mathcal{T})$ to denote the ground instance $(\mu_1(T_1), \ldots, \mu_n(T_n))$ of $\mathcal{R}$.

A *partially closed c-instance* $\mathcal{T}$ represents a set of partially closed ground instances, denoted by $\mathsf{Mod}(\mathcal{T}, D_m, V)$:

$$\{\mu(\mathcal{T}) \mid \mu \text{ is a valuation}, (\mu(\mathcal{T}), D_m) \models V\}.$$

We write $\mathsf{Mod}(\mathcal{T}, D_m, V)$ as $\mathsf{Mod}(\mathcal{T})$ when $D_m$ and $V$ are clear from the context.

In the sequel we consider only *c*-instances $\mathcal{T}$ for which $\mathsf{Mod}(\mathcal{T})$ is nonempty. As will be seen in Section 3, it is decidable to determine whether $\mathsf{Mod}(\mathcal{T})$ is empty.

Databases under the CWA or the OWA are special cases of partially closed *c*-instances. A *c*-instance $\mathcal{T}$ is open-world in the absence of master data and CCs. It is closed-world if master data is a possible world represented by $\mathcal{T}$.

**Relative completeness**. *Relative to* $(D_m, V)$, a partially closed *c*-instance $\mathcal{T}$ is said to be

- *strongly complete for $Q$* if *for each* $\mathcal{I} \in \mathsf{Mod}(\mathcal{T})$ and *for each* $\mathcal{I}' \in \mathsf{Ext}(\mathcal{I})$, $Q(\mathcal{I}) = Q(\mathcal{I}')$;
- *weakly complete for $Q$* if $\bigcap_{\mathcal{I} \in \mathsf{Mod}(\mathcal{T})} Q(\mathcal{I}) = \bigcap_{\mathcal{I} \in \mathsf{Mod}(\mathcal{T}), \mathcal{I}' \in \mathsf{Ext}(\mathcal{I})} Q(\mathcal{I}')$, or for all $\mathcal{I} \in \mathsf{Mod}(\mathcal{T})$, $\mathsf{Ext}(\mathcal{I}) = \emptyset$; and
- *completable for $Q$* if *there exists* $\mathcal{I} \in \mathsf{Mod}(\mathcal{T})$ such that for each $\mathcal{I}' \in \mathsf{Ext}(\mathcal{I})$, $Q(\mathcal{I}) = Q(\mathcal{I}')$.

Intuitively, (a) $\mathcal{T}$ is strongly complete if no matter how missing values in $\mathcal{T}$ are filled in, it yields a ground instance relatively complete for $Q$; (b) $\mathcal{T}$ is weakly complete if the certain answer to $Q$ over all partially closed extensions of $\mathcal{T}$ can be found in $\mathcal{T}$; and (c) $\mathcal{T}$ is completable if there exists a way to instantiate missing values in $\mathcal{T}$ and make it a ground instance relatively complete for $Q$.

**Example 2.3:** Consider the *c*-instance $T$ of Fig. 1, $D_m$ and $Q_1$ of Example 1.1 and the set $V$ of CCs of Example 2.1. Then $T$ is strongly complete for $Q_1$ relative to $(D_m, V)$. Indeed, by the FD $\phi$ encoded as CCs in $V$, for any valuation $\mu$ of $T$, $Q_1(\mu(T))$ returns a single tuple ($\mathsf{str}$ = '3 Elm'), and the answer does not change for any instance in $\mathsf{Ext}(\mu(T))$.

Now consider query $Q_4$ to find the names of Edi patients born in 2000. Suppose that $t_m^1$ and $t_m^2$ are the only patients in $D_m$ born in 2000, where $t_m^1$ = (John, 3 Elm, EH8 9AB, 2000) and $t_m^2$ = (Bob, 3 Elm, EH8 9AB, 2000). Then relative to $(D_m, V)$, $T$ is completable for $Q_4$, since there exists a valuation $\mu$ of $T$ such that $\mu(T)$ is complete, *i.e.,* when $\mu(x)$ = Bob and $\mu(z) = 2000$. It is also weakly complete, since the certain answer ($\mathsf{name}$ = 'John') can already be found over $\mathsf{Mod}(T)$. However, $T$ is not strongly complete for $Q_4$. Indeed, consider $\mu'(T)$ with $\mu'(x)$ = John and $\mu'(z) = 2000$, and $\mu(T)$ defined as before. Then, clearly, $\mu'(T) \subseteq \mu(T)$ and moreover, $Q_4(\mu'(T))$ only returns John whereas $Q_4(\mu(T))$ returns both John and Bob. □

Observe the following. (a) If $\mathcal{T}$ is strongly complete, then it is both weakly complete and completable. (b) A ground instance $\mathcal{I}$ is a *c*-instance without variables and conditions. It is strongly complete and completable for a query $Q$ iff $\mathcal{I}$ is relatively complete for $Q$ as defined in Section 2.1. However, $\mathcal{I}$ may be weakly complete but not relatively complete.

We use $\mathsf{RCQ}(Q, D_m, V)$ to denote the set of all strongly complete $c$-instances of $\mathcal{R}$ for $Q$ *w.r.t.* $(D_m, V)$ (resp. completable, weakly complete when it is clear from the context). Since $\mathcal{R}$ is always clear from the context, we do not include it as a parameter for $\mathsf{RCQ}$.

**Minimal complete databases**. To decide what data should be collected in a database to answer a query $Q$, we want to identify a minimal amount of information that is complete for $Q$. For this, we use a notion of minimality.

A ground instance $\mathcal{I}$ is a *minimal* instance complete for a query $Q$ relative to $(D_m, V)$ if it is in $\mathsf{RCQ}(Q, D_m, V)$ and moreover, for any $\mathcal{I}' \subset \mathcal{I}$, $\mathcal{I}'$ is not in $\mathsf{RCQ}(Q, D_m, V)$.

A $c$-instance $\mathcal{T}$ is a *minimal* $c$-instance completable (resp. strongly complete) for $Q$ relative to $(D_m, V)$ if there exists $\mathcal{I} \in \mathsf{Mod}(\mathcal{T})$ (resp. for all $\mathcal{I} \in \mathsf{Mod}(\mathcal{T})$) such that $\mathcal{I}$ is a *minimal* instance complete for a query $Q$.

To define minimal instances in the weak model, we write $(T, \xi) \subset (T', \xi')$ if $T \subset T'$ and $\xi$ is the restriction of $\xi'$ on $T$. For $\mathcal{T} = (T_1, \ldots, T_n)$ and $\mathcal{T}' = (T'_1, \ldots, T'_n)$, we write $\mathcal{T} \subset \mathcal{T}'$ if $T_i \subseteq T'_i$ for all $i \in [1, n]$, and $T_j \subset T'_j$ for some $j$.

A database $\mathcal{T}$ is a *minimal instance weakly complete* for $Q$ relative to $(D_m, V)$ if $\mathcal{T}$ is in $\mathsf{RCQ}(Q, D_m, V)$ and there exists no $\mathcal{T}' \subset \mathcal{T}$ such that $\mathcal{T}'$ is in $\mathsf{RCQ}(Q, D_m, V)$. Note that $\mathcal{T}'$ can be either a $c$-instance or a ground instance.

**Example 2.4:** Recall $D_m, V$ and $Q_2$ from Example 2.2. Then as argued there, a ground instance $D$ is minimally strongly complete for $Q_2$ as long as $D$ consists of a single tuple $t$ with $t[\mathsf{zip}]$ = 'EH1 3CD'. This tells us that minimal complete instances may not be unique. In contrast, $D$ is a minimal instance weakly complete for $Q_2$ if $D$ is empty.

As shown in Example 2.3, the $c$-instance $T$ of Fig. 1 is strongly complete for $Q_1$. However, it is not minimal: removing $t_2, t_3$ from $T$ yields a smaller complete database. □

### 2.3 Deciding Relative Completeness

We study four problems associated with relative complete databases, parametrized with a query language $\mathcal{L}_Q$.

| | |
|---|---|
| $\mathsf{RCDP}(\mathcal{L}_Q)$: | The *relatively complete database* problem. |
| INPUT: | A query $Q$ in $\mathcal{L}_Q$, master data $D_m$, a set $V$ of CCs, and a partially closed $c$-instance $\mathcal{T}$ *w.r.t.* $(D_m, V)$. |
| QUESTION: | Is $\mathcal{T}$ in $\mathsf{RCQ}(Q, D_m, V)$? |

That is, does $\mathcal{T}$ have complete information to answer $Q$?

| | |
|---|---|
| $\mathsf{RCQP}(\mathcal{L}_Q)$: | The *relatively complete query* problem. |
| INPUT: | $Q, D_m$ and $V$ as in $\mathsf{RCDP}$. |
| QUESTION: | Is $\mathsf{RCQ}(Q, D_m, V)$ nonempty? |

It is to determine whether there exists a $c$-instance with complete information to answer $Q$.

| | |
|---|---|
| $\mathsf{MinP}(\mathcal{L}_Q)$: | The *minimality* problem. |
| INPUT: | $Q, D_m, V$ and $\mathcal{T}$ as in $\mathsf{RCDP}$. |
| QUESTION: | Is $\mathcal{T}$ a minimal $c$-instance complete for $Q$ relative to $(D_m, V)$? |

This asks whether $\mathcal{T}$ is a minimal-size database complete for $Q$, *i.e.,* removing any tuple from $\mathcal{T}$ makes it incomplete.

| | |
|---|---|
| $\mathsf{BdnP}(\mathcal{L}_Q)$: | The *boundedness* problem. |
| INPUT: | A number $K$, and $Q, D_m, V$ as in $\mathsf{RCDP}$. |
| QUESTION: | Does there exist a $c$-instance $\mathcal{T}$ such that $\mathcal{T}$ is in $\mathsf{RCQ}(Q, D_m, V)$ and $\lvert\mathcal{T}\rvert \leq k$? |

Here $|\mathcal{T}|$ denotes the cardinality of $\mathcal{T}$, *i.e.,* the number of tuple templates in $\mathcal{T}$. This problem asks whether there exists a database of a bounded size, *i.e.,* with at most $K$ tuples, that carries complete information to answer a query.

We study these problems when $\mathcal{L}_Q$ ranges over the following query languages (see, *e.g.,* [2], for the details):

- CQ, the class of conjunctive queries built up from atomic formulas, *i.e.,* relation atoms in the schema $\mathcal{R}$, equality ($=$) and inequality ($\neq$), by closing under conjunction $\wedge$ and existential quantification $\exists$;
- UCQ, union of conjunctive queries of the form $Q_1 \cup \cdots \cup Q_k$, where for each $i \in [1, k]$, $Q_i$ is in CQ;
- $\exists$FO$^+$, first-order logic (FO) queries built from atomic formulas, by closing under $\wedge$, *disjunction* $\vee$ and $\exists$;
- FO queries built from atomic formulas using $\wedge$, $\vee$, *negation* $\neg$, $\exists$ and universal quantification $\forall$; and
- FP, an extension of $\exists$FO$^+$with an inflational fixpoint operator, *i.e.,* queries defined as a collection of rules $p(\vec{x}) \leftarrow p_1(\vec{x}_1), \ldots, p_m(\vec{x}_m)$, where each $p_i$ is either an atomic formula or an IDB predicate.

We also investigate the special case for *ground instances*. In this setting, $\mathsf{RCQP}(\mathcal{L}_Q)$ is to decide, given $Q$ in $\mathcal{L}_Q$, $D_m$ and $V$, whether there exists a ground instance in $\mathsf{RCQ}(Q, D_m, V)$. Similarly $\mathsf{RCDP}(\mathcal{L}_Q)$, $\mathsf{MinP}(\mathcal{L}_Q)$ and $\mathsf{BdnP}(\mathcal{L}_Q)$ can be stated for ground instances.

We study these problems when $\mathsf{RCQ}(Q, D_m, V)$ denotes the set of instances that are strongly complete, weakly complete or completable, in Sections 4, 5 and 6, respectively.

## 3. Analysis of Partially Closed Databases

Before we study the decision problems for relative completeness, we investigate some basic problems in connection with integrity constraints and partially closed databases.

**The impact of integrity constraints**. Several classes of constraints have been used to specify data consistency, notably denial constraints and conditional functional dependencies (CFDs) (see [6, 8] for surveys). As shown in [9], denial constraints and CFDs can be expressed as CCs of Section 2. Hence we can enforce both relative information completeness and data consistency using those CCs.

One might want a more powerful class $\mathcal{C}$ of constraints to specify the consistency. More specifically, one may want to require a partially closed database $\mathcal{I}$ to satisfy a set $\Theta$ of constraints in $\mathcal{C}$, in addition to being bounded by master data $D_m$ via a set $V$ of CCs. Similarly, partially closed extensions of $\mathcal{I}$ are also required to satisfy the additional $\Theta$.

However, the choice of constraints has an immediate impact on the analysis of relative completeness. When $\mathcal{C}$ consists of, *e.g.,*FDs and inclusion dependencies (INDs), both $\mathsf{RCDP}(\mathcal{L}_Q)$ and $\mathsf{RCQP}(\mathcal{L}_Q)$ are beyond reach in practice for any language $\mathcal{L}_Q$, even in the absence of missing values, when the relative completeness of Section 2.1 is concerned.

**Proposition 3.1:** In the presence of FDs and INDs, $\mathsf{RCDP}$ and $\mathsf{RCQP}$ for ground instances are undecidable for CQ. □

Proof. The undecidability is verified by reduction from the implication problem for INDs and FDs taken together, which is known to be undecidable (cf. [2]). It is undecidable even when only keys and foreign keys are considered, for which the implication problem is undecidable [10]. □

This suggests that we consider integrity constraints that are expressible as CCs, to focus on the complexity incurred

by the analysis of relative completeness rather than by integrity constraints. As remarked earlier, the CCs are powerful enough to express constraints often used in data cleaning.

**Reasoning about $c$-instances**. As remarked earlier, the analysis of relative completeness requires decision procedures for determining some basic problems in connection with partially closed $c$-instances, which are stated as follows.

- *The consistency problem* is to determine, given master data $D_m$, a set $V$ of CCs and a $c$-instance $\mathcal{T}$, whether $\mathsf{Mod}(\mathcal{T}, D_m, V)$ is nonempty, *i.e.,* whether $\mathcal{T}$ makes sense.
- *The extensibility problem* is to determine, given $D_m$, $V$ and a ground instance $\mathcal{I}$, whether $\mathsf{Ext}(\mathcal{I}, D_m, V)$ is nonempty, *i.e.,* whether $\mathcal{I}$ can be expanded without violating $V$.

**Proposition 3.2:** The consistency and extensibility problems are both $\Sigma_2^p$-complete. The complexity is unchanged even in the absence of local conditions in $c$-instances. □

Proof. The upper bound for consistency (resp. extensibility) is proved by giving a $\Sigma_2^p$ algorithm for checking the non-emptiness of $\mathsf{Mod}(\mathcal{T})$ (resp. $\mathsf{Ext}(\mathcal{I})$).

The $\Sigma_2^p$ lower bounds are verified by reduction from the $\exists^*\forall^*$3SAT problem, which is $\Sigma_2^p$-complete (cf. [26]). The problems are already $\Sigma_2^p$-hard for $c$-instances (or ground) with a fixed number of tuples, without local conditions. □

We should remark that these problems do not increase the complexity bounds on RCDP, RCQP, MinP and BdnP.

# 4. Strong Relative Information Completeness

We now study RCDP, RCQP, MinP and BdnP for strongly relatively complete databases. In the strong completeness model, we focus on databases in which neither missing values nor missing tuples prevent them from having complete information for answering queries relative to master data.

We establish complexity bounds on these problems for $c$-instances. For ground instances, we provide complexity results not given in [9], *i.e.,* for $\mathsf{MinP}(\mathcal{L}_Q)$ and $\mathsf{BdnP}(\mathcal{L}_Q)$, and for the cases of $\mathsf{RCQP}(\mathcal{L}_Q)$ left open in [9].

Our main conclusion about the strong model is that missing values make our lives harder, but not too much.

*(1)* $\mathsf{RCDP}(\mathcal{L}_Q)$. This problem is to decide whether a given database is relatively complete for a query. It is known [9] that for ground instances, $\mathsf{RCDP}(\mathcal{L}_Q)$ is undecidable when $\mathcal{L}_Q$ is FO or FP, and it is $\Pi_2^p$-complete when $\mathcal{L}_Q$ ranges over CQ, UCQ and $\exists$FO$^+$. The result below tells us that the presence of missing values complicates the analysis: even $\mathsf{RCDP}$(CQ) becomes $\Pi_3^p$-complete for $c$-instances.

In practice, master data $D_m$ and the set $V$ of CCs are often predefined and fixed, and only databases and user queries vary. One might think that RCDP would become simpler in this setting. Unfortunately, this is not the case: the complexity bounds remain intact when $D_m$ and $V$ are fixed.

**Theorem 4.1:** In the strong model, $\mathsf{RCDP}(\mathcal{L}_Q)$ is

- undecidable when $\mathcal{L}_Q$ is either FO or FP, and
- $\Pi_3^p$-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$,

for $c$-instances. The complexity bounds remain unchanged when master data $D_m$ and the set $V$ of CCs are fixed. □

Proof. (1) Note that $\mathsf{RCDP}$(FO) and $\mathsf{RCDP}$(FP) are undecidable for ground instances [9], which are also $c$-instances.

We provide an alternative proof of the undecidability of $\mathsf{RCDP}$(FP) by reduction from *the satisfiability problem for FP in the presence of FDs*. Given a FP query $p$ and a set $\Theta$ of FDs, it is to decide whether there exists a database $D$ such that $D \models \Theta$ and $p(D) \neq \emptyset$. The undecidability of the problem was claimed in [19]. We show a stronger result: the problem is already undecidable when the set of FDs is *fixed*. This is verified by reduction from the emptiness problem for deterministic finite 2-head automata, which is undecidable [29].

(2) For the $\Pi_3^p$-completeness, we first show that $\mathsf{RCDP}$(CQ) is already $\Pi_3^p$-hard for $c$-instances, by reduction from $\forall^*\exists^*\forall^*$3SAT, a $\Pi_3^p$-complete problem (cf. [26]). We then show that $\mathsf{RCDP}(\exists\text{FO}^+)$ is in $\Pi_3^p$, by providing a $\Sigma_3^p$ algorithm for deciding whether a $c$-instance is *not* relatively complete for an $\exists$FO$^+$query. The algorithm is based on a *small model property* on such $c$-instances, which is in turn established by developing a characterization of such $c$-instances.

The lower bound proofs require fixed $D_m$ and $V$ only. □

*(2)* $\mathsf{RCQP}(\mathcal{L}_Q)$. This is to determine whether a given query can find a relatively complete database at all. When it comes to $\mathsf{RCQP}(\mathcal{L}_Q)$, one does not have to worry about missing values in the strong model. Indeed, $\mathsf{RCQP}(\mathcal{L}_Q)$ for $c$-instances and its counterpart for ground instances coincide.

**Lemma 4.2:** In the strong model, for any schema $\mathcal{R}$, query $Q$, master data $D_m$, any set $V$ of CCs and any number $K$, there exists a $c$-instance $\mathcal{T}$ of $\mathcal{R}$ such that $|\mathcal{T}| \leq K$ and $\mathcal{T} \in \mathsf{RCQ}(Q, D_m, V)$ iff there exists a ground instance $\mathcal{I}$ of $\mathcal{R}$ such that $|\mathcal{I}| \leq K$ and $\mathcal{I} \in \mathsf{RCQ}(Q, D_m, V)$. □

As a result one only needs to consider $\mathsf{RCQP}(\mathcal{L}_Q)$ for ground instances. Nevertheless, the complexity bounds on $\mathsf{RCQP}(\mathcal{L}_Q)$ were left open in [9] when $\mathcal{L}_Q$ is FO or FP, for ground instances. Indeed, $\mathsf{RCQP}(\mathcal{L}_Q)$ was shown undecidable there by using CCs expressed as fixed FO or FP queries. Below we settle these cases.

**Corollary 4.3:** In the strong model, $\mathsf{RCQP}(\mathcal{L}_Q)$ is

- undecidable when $\mathcal{L}_Q$ is FO or FP; and
- NEXPTIME-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$,

for $c$-instances. The complexity bounds remain unchanged when $D_m$ and $V$ are fixed. □

Proof. (1) We show that $\mathsf{RCQP}$(FO) is undecidable by reduction from the satisfiability problem for FO queries, which is undecidable (cf. [2]). For FP, the undecidability is proved by reduction from the satisfiability problem for FP in the presence of *fixed* FDs, for which the undecidability was shown in the proof of Theorem 4.1.

(2) It is known [9] that $\mathsf{RCQP}(\exists\text{FO}^+)$ is in NEXPTIME and $\mathsf{RCQP}$(CQ) is NEXPTIME-hard for ground instances. By Lemma 4.2 these results remain intact for $c$-instances.

The proofs use fixed $D_m$ and $V$ for the lower bounds. □

*(3)* $\mathsf{MinP}(\mathcal{L}_Q)$. This is to decide whether a database is relatively complete and moreover, does not contain excessive data. The lemma below tells us how to check this.

**Lemma 4.4:** In the strong model, for any ground instance $I$, query $Q$, master data $D_m$, and any set $V$ of CCs, (a) if $(I, D_m) \models V$ then for any $I' \subset I$, $(I', D_m) \models V$, and (b) if $I$ is in $\mathsf{RCQ}(Q, D_m, V)$, then $I$ is not minimal iff there exists a tuple $t \in I$ such that $I \setminus \{t\} \in \mathsf{RCQ}(Q, D_m, V)$. □

PROOF. The lemma can be readily verified based on the monotonicity of CQ queries that define CCs, and by the definition of relative strong completeness. □

Capitalizing on this lemma, below we provide complexity bounds on $\mathsf{MinP}(\mathcal{L}_Q)$, both for $c$-instances and for ground instances. Here the presence of missing values again makes the problem a little harder: $\mathsf{MinP}(\text{CQ})$ is $\Delta_3^p$-complete for ground instances, but it is $\Pi_3^p$-complete for $c$-instances.

**Theorem 4.5:** In the strong model,

- when $\mathcal{L}_Q$ is FO or FP, $\mathsf{MinP}(\mathcal{L}_Q)$ is undecidable both for ground instances and for $c$-instances;
- when $\mathcal{L}_Q$ is CQ, UCQ or $\exists\text{FO}^+$, $\mathsf{MinP}(\mathcal{L}_Q)$ is
  - $\Pi_3^p$-complete for $c$-instances, and
  - $\Delta_3^p$-complete for ground instances.

The complexity is unchanged when $D_m$ and $V$ are fixed. □

PROOF. (1) The undecidability for FO and FP is verified by extending the proofs for their counterparts in Corollary 4.3 and Theorem 4.1, respectively.

(2) For $c$-instances, we first show that $\mathsf{MinP}(\text{CQ})$ is $\Pi_3^p$-hard by reduction from $\forall^*\exists^*\forall^*$3SAT. We then show that MinP is in $\Pi_3^p$ for $\exists\text{FO}^+$, by giving a $\Sigma_3^p$ algorithm for checking whether a $c$-instance is *not* minimally complete. It leverages Lemma 4.4 and the characterization of relatively complete $c$-instances given in the proof of Theorem 4.1.

(3) For ground instances, we first show that $\mathsf{MinP}(\text{CQ})$ is $\Delta_3^p$-hard by reduction from the MSA($\exists^*\forall^*$3SAT) problem, which is shown $\Delta_3^p$-complete [16] by its connection with a polynomial 2-alternating Turing machine with the max and min operators. We then prove that $\mathsf{MinP}(\exists\text{FO}^+)$ is in $\Delta_3^p$ by giving a decision procedure that invokes a $\Sigma_2^p$ oracle polynomially many times, taking advantage of Lemma 4.4.

The lower-bound proofs only use fixed $D_m$ and $V$. □

*(4)* $\mathsf{BdnP}(\mathcal{L}_Q)$. Given a number $K$ and a query $Q$, this problem is to decide whether there exists a database that consists of at most $K$ tuples and is relatively complete for $Q$. We give its complexity bounds below, which tell us that the bounds on the problem for $c$-instances are the same as their counterparts for ground instances, for any $K$ and $Q$.

In contrast to the results given above, there are subtle differences between CQ and UCQ in the minimality analysis. It takes a single tuple ($K \geq 1$) to show that $\mathsf{BdnP}(\text{UCQ})$ is $\Sigma_3^p$-hard, while it requires 14 tuples ($K \geq 14$) for $\mathsf{BdnP}(\text{CQ})$.

**Theorem 4.6:** In the strong model, $\mathsf{BdnP}(\mathcal{L}_Q)$ is

- undecidable if $\mathcal{L}_Q$ is FO or FP, for any $K \geq 0$,
- $\Sigma_3^p$-complete when $\mathcal{L}_Q$ is CQ for any $K \geq 14$, and
- $\Sigma_3^p$-complete if $\mathcal{L}_Q$ is UCQ or $\exists\text{FO}^+$, for any $K \geq 1$,

both for $c$-instances and for ground instances. □

PROOF. From Lemma 4.2 it follows that $\mathsf{BdnP}(\mathcal{L}_Q)$ for $c$-instances and $\mathsf{BdnP}(\mathcal{L}_Q)$ for ground instances coincide. Hence it suffices to focus on ground instances.

(1) For FO and FP, the undecidability is verified by reduction from FO satisfiability and the satisfiability of FP queries in the presence of fixed FDs, respectively.

(2) We show that $\mathsf{BdnP}(\text{CQ})$ is $\Sigma_3^p$-hard by reduction from the $\exists^*\forall^*\exists^*$3SAT problem, which is $\Sigma_3^p$-complete (cf. [26]). The reduction needs 14 tuples to encode disjunction and negation, which are not supported by CQ. With disjunction in UCQ, $\mathsf{BdnP}(\text{UCQ})$ is verified $\Sigma_3^p$-hard by a different reduction from $\exists^*\forall^*\exists^*$3SAT, which requires one tuple only.

We then show that $\mathsf{BdnP}(\exists\text{FO}^+)$ is in $\Sigma_3^p$ by giving a $\Sigma_3^p$ algorithm. The algorithm first guesses a ground instance $\mathcal{I}$ of at most $K$ tuples (by Lemma 4.2), and then calls an oracle to check whether $\mathcal{I}$ is not relatively complete. The latter can be done in $\Sigma_2^p$ [9]. Compared to RCQP, BdnP has a significantly lower complexity because for BdnP, we only need to inspect instances with a bounded size. □

# 5. Weak Relative Information Completeness

We next investigate RCDP, RCQP, MinP and BdnP for weakly complete databases, *i.e.,* databases from which one can find the certain answers to a query over their partially closed extensions. In the weak completeness model, none of these problems has been studied, for either $c$-instances or ground instances. We provide their complexity bounds here.

Compared to their counterparts in the strong model, the complexity results in the weak model are more diverse. On one hand, the certain-answer semantics simplifies the analysis of some problems, *e.g.,* all these problems become decidable for FP, in contrast to their undecidability in the strong model. On the other hand, it makes certain problems harder, *e.g.,* MinP becomes $\Pi_4^p$-complete for UCQ, as opposed to $\Pi_3^p$. In addition, some problems even have different bounds for CQ and UCQ, *e.g.,* MinP and BdnP.

*(1)* $\mathsf{RCDP}(\mathcal{L}_Q)$. As opposed to Theorem 4.1, in the weak completeness model RCDP is decidable for FP. In addition, RCDP for $c$-instances and RCDP for ground instances are both $\Pi_p^3$-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists\text{FO}^+$, while their counterparts in the strong model are $\Pi_p^3$-complete (Theorem 4.1) and $\Pi_p^2$-complete [9], respectively.

**Theorem 5.1:** In the weak model, $\mathsf{RCDP}(\mathcal{L}_Q)$ is

- undecidable when $\mathcal{L}_Q$ is FO,
- coNEXPTIME-complete when $\mathcal{L}_Q$ is FP, and
- $\Pi_p^3$-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists\text{FO}^+$,

for $c$-instances and for ground instances. The complexity bounds remain unchanged when $D_m$ and $V$ are fixed. □

PROOF. (1) We show that it is already undecidable to decide whether an empty database is weakly complete for FO queries, by reduction from the FO satisfiability problem.

(2) For FP, we show that RCDP is coNEXPTIME-hard for ground instances by reduction from the SUCCINCT-TAUT problem, the complement of SUCCINCT-CIRCUIT-SAT that is NEXPTIME-complete (cf. [26]). We then provide an NEXPTIME algorithm to check whether a $c$-instance is *not* weakly complete, by leveraging the certain-answer semantics and the monotonicity of FP. Hence it is coNEXPTIME-complete for $c$-instances and for ground instances.

(3) We show the $\Pi_p^3$-hardness of $\mathsf{RCDP}(\text{CQ})$ for ground instances by reduction from $\forall^*\exists^*\forall^*$3SAT, and give a $\Sigma_p^3$ algorithm to check whether a $c$-instance is *not* weakly complete for an $\exists\text{FO}^+$query. The reduction uses fixed $D_m$ and $V$. □

*(2)* $\mathsf{RCQP}(\mathcal{L}_Q)$. Recall that in the strong model, RCQP for $c$-instances is equivalent to RCQP for ground instances, as verified by Lemma 4.2. However, the example below tells us that it is no longer the case in the weak completeness model.

**Example 5.1:** Consider an FO query $Q$ defined on a pair of relations: $Q(I_1, I_2) = \{(a)\}$ if $I_1 \subseteq I_2$, and it is $\{(b)\}$

otherwise, where $a$ and $b$ are distinct. For empty $D_m$ and $V$, no ground instances are in $\mathsf{RCQ}(Q, D_m, V)$ since $Q(I_1, I_2) \neq \emptyset$ for all $(I_1, I_2)$ while $\bigcap_{\mathcal{I}' \in \mathsf{Ext}(I_1,I_2)} Q(\mathcal{I}') = \emptyset$. In contrast, there exists a $c$-instance $\mathcal{T} = (T_1, T_2)$ in $\mathsf{RCQ}(Q, D_m, V)$, where $T_1$ and $T_2$ are singleton $c$-tables without conditions, each having a tuple with variables only. Indeed, $Q(\mathcal{T}) = \bigcap_{\mathcal{I} \in \mathsf{Mod}(\mathcal{T})} Q(\mathcal{I}) = \emptyset = \bigcap_{\mathcal{I} \in \mathsf{Mod}(\mathcal{T}), \mathcal{I}' \in \mathsf{Ext}(\mathcal{I})} Q(\mathcal{I}')$. □

This tells us that from the undecidability of $\mathsf{RCQP}$(FO) for ground instances we cannot conclude the undecidability for $c$-instances. Nevertheless, $\mathsf{RCQP}(\mathcal{L}_Q)$ becomes trivially decidable when $\mathcal{L}_Q$ is FP, CQ, UCQ or $\exists$FO$^+$, for $c$-instances and for ground instances, in contrast to Theorem 4.3.

**Theorem 5.2:** In the weak model, $\mathsf{RCQP}(\mathcal{L}_Q)$ is

- undecidable for ground instances if $\mathcal{L}_Q$ is FO, and
- decidable in $O(1)$-time for $c$-instances and for ground instances when $\mathcal{L}_Q$ is FP, CQ, UCQ or $\exists$FO$^+$.

The complexity is unchanged when $D_m$ and $V$ are fixed. □

Proof. (1) The undecidability of $\mathsf{RCQP}$(FO) is verified for ground instances by reduction from the satisfiability problem for FO. The reduction uses neither $D_m$ nor $V$.

(2) We show that for any query $Q$ in FP, master data $D_m$ and any set $V$ of CCs, $\mathsf{RCQ}(Q, D_m, V) \neq \emptyset$. Indeed, we can always construct a ground instance (and hence, a $c$-instance) weakly complete for $Q$ relative to $(D_m, V)$. The construction leverages the monotonicity of FP. □

*(3)* $\mathsf{MinP}(\mathcal{L}_Q)$. Lemma 4.4 no longer holds in the weak completeness model, *i.e.,* to decide whether an instance $I$ is minimal, it does not suffice to inspect $I \setminus \{t\}$ only.

**Example 5.2:** Consider a CQ query $Q$ defined on a pair of unary relations $(R_1, R_2)$: $Q(x) = (R_1(y) \wedge R_2(z) \wedge x = a)$. That is, on an instance $(I_1, I_2)$ of $(R_1, R_2)$, $Q$ returns $\{(a)\}$ if $I_1$ and $I_2$ are both nonempty. Consider an instance $\mathcal{I}_0 = (\{(0)\}, \{(1)\})$, an empty set $V$ of CCs and any master data $D_m$. Then $\mathcal{I}_0$ is weakly complete for $Q$ relative to $(D_m, V)$. Nevertheless, it is not minimal: the empty instance $(\emptyset, \emptyset)$ of $(R_1, R_2)$ is also in $\mathsf{RCQ}(Q, D_m, V)$. However, removing one tuple from $\mathcal{I}_0$ does not make it a weakly complete instance, *i.e.,* a counterexample to the minimality of $\mathcal{I}_0$ cannot be found by removing only one tuple from $\mathcal{I}_0$. □

In the weak model, the minimality analysis is quite different from its counterpart in the strong model (Theorem 4.5). (a) The absence of missing values does not simplify the analysis, as opposed to their counterparts in the strong model ($\Delta_3^p$ for ground instances vs. $\Pi_3^p$ for $c$-instances). (b) It is much easier to check $\mathsf{MinP}$(CQ) than $\mathsf{MinP}$(UCQ) (coDP-complete vs. $\Pi_p^4$-complete), whereas in the strong model, $\mathsf{MinP}$(CQ) and $\mathsf{MinP}$(UCQ) have the same complexity.

**Theorem 5.3:** In the weak model, $\mathsf{MinP}(\mathcal{L}_Q)$ is

- undecidable when $\mathcal{L}_Q$ is FO,
- coNEXPTIME-complete when $\mathcal{L}_Q$ is FP,
- $\Pi_p^4$-complete when $\mathcal{L}_Q$ is UCQ or $\exists$FO$^+$, and
- coDP-complete when $\mathcal{L}_Q$ is CQ,

both for $c$-instances and for ground instances. □

Proof. (1) We show that $\mathsf{MinP}$(FO) is already undecidable for fixed ground instances, as for Theorem 5.1 (1).

(2) We show that $\mathsf{MinP}$(FP) is coNEXPTIME-hard for ground instances again by reduction from the SUCCINCT-TAUT problem. For the upper bound, we develop an NEXPTIME algorithm that, given a $c$-instance $\mathcal{T}$, a FP query $Q$, master data $D_m$ and CCs $V$, returns true if either $\mathcal{T}$ is *not* weakly complete for $Q$ relative to $(D_m, V)$, or there exists a $c$-instance smaller than $\mathcal{T}$ that is in $\mathsf{RCQ}(Q, D_m, V)$.

(3) We first show that $\mathsf{MinP}$(UCQ) is $\Pi_p^4$-hard for ground instances, by reduction from the $\forall^*\exists^*\forall^*\exists^*$3SAT problem, which is known to be $\Pi_p^4$-complete (cf. [26]). The reduction makes heavy use of disjunction in UCQ. We then provide a $\Sigma_p^4$ algorithm for determining whether a $c$-instance is *not* a minimal instance weakly complete for $\exists$FO$^+$queries, calling a $\Sigma_p^3$ oracle for completeness checking (Theorem 5.1).

(4) For CQ queries in the weak completeness model, minimally complete instances are rather restrictive. This is verified by the following lemma. For any CQ query $Q$, master data $D_m$ and any set $V$ of CCs, (a) there always exists a minimal $c$-instance $\mathcal{T}$ in $\mathsf{RCQ}(Q, D_m, V)$ such that either $\mathcal{T}$ is the empty instance $\mathcal{T}_\emptyset$, or $\mathcal{T}$ is a singleton set; and (b) if $\mathcal{T}_\emptyset$ is not in $\mathsf{RCQ}(Q, D_m, V)$, then any singleton $\mathcal{T}'$ (with nonempty $\mathsf{Mod}(\mathcal{T}', D_m, V)$) is in $\mathsf{RCQ}(Q, D_m, V)$.

By this lemma we only need to consider those $c$-instances $\mathcal{T}$ such that either $\mathcal{T} = \emptyset$ or $|\mathcal{T}| = 1$. Moreover, when $|\mathcal{T}| \leq 1$, the problem for determining whether $\mathcal{T}$ is a relatively complete minimal instance is reduced to the problem for deciding whether $\mathcal{T}_\emptyset$ is in $\mathsf{RCQ}(Q, D_m, V)$. We give a coDP algorithm to check the latter. From this it follows that the minimality analysis for CQ is in coDP in the weak model.

For the lower bound, we show that it is already coDP-hard to decide whether $\mathcal{T}_\emptyset$ is in $\mathsf{RCQ}(Q, D_m, V)$, and hence is minimally complete. This is verified by reduction from the complement of the SAT-UNSAT problem. The latter is to decide whether for a pair $(\phi, \phi')$ of 3SAT instances, $\phi$ is satisfiable and $\phi'$ is not, which is DP-complete (cf. [26]). □

*(4)* $\mathsf{BdnP}(\mathcal{L}_Q)$. Compared to Theorem 4.6, the impact of the certain query-answer semantics is also apparent on $\mathsf{BdnP}$.

**Theorem 5.4:** In the weak model, $\mathsf{BdnP}(\mathcal{L}_Q)$ is

- undecidable for FO and any $K \geq 0$,
- coNEXPTIME-complete for FP and any $K \geq 1$,
- $\Sigma_4^p$-complete for UCQ, $\exists$FO$^+$and any $K \geq 1$, and
- coDP-complete for CQ and any $K \geq 0$,

both for $c$-instances and for ground instances. □

Proof. (1) We show that for any $K \geq 0$, $\mathsf{BdnP}$ is undecidable for FO by reduction from FO satisfiability.

(2) For FP and any $K \geq 1$, we show that $\mathsf{BdnP}$ is coNEXPTIME-hard by reduction from the SUCCINCT-TAUT problem. We then give an algorithm that inspects $c$-instances of at most $K$ tuples, and invokes a coNEXPTIME oracle to check whether such instances are weakly complete for a FP query. The number of such instances is bounded by an exponential number, based on a small model property.

Note that the proof of Theorem 5.2 (2) given above does not guarantee that the complete instances constructed have a bounded size. More checking is required for deciding the existence of a complete instance with a given size, and hence, the higher complexity on $\mathsf{BdnP}$ than on $\mathsf{RCQP}$.

(3) For ground instances and any $K \geq 1$, we show that $\mathsf{BdnP}$(UCQ) is $\Sigma_p^4$-hard by reduction from $\exists^*\forall^*\exists^*\forall^*$3SAT, a known $\Sigma_p^4$-complete problem (cf. [26]). We then give a $\Sigma_p^4$

algorithm for deciding whether there exists a weakly complete $c$-instance of size at most $K$. This invokes a $\Sigma_p^3$ oracle to check various $c$-instances of size at most $K$, and hence, has a higher complexity than RCDP.

(4) The lemma given in the proof of Theorem 5.3 (4) tells us that for any $K \geq 1$ the answer to BdnP(CQ) is positive. When $K = 0$, BdnP(CQ) is equivalent to deciding whether the empty instance is relatively complete, which is coDP-complete as verified by the proof of Theorem 5.3 (4). □

## 6. Relatively Completable Databases

Finally we investigate RCDP, RCQP, MinP and BdnP for completable $c$-instances, *i.e.,* databases that can be made relatively complete when their missing values are correctly instantiated. In this model we provide complexity results on these problems, for various query languages. The results tell us that missing values complicate the analysis of these problems, to an extent. As opposed to their counterparts in the weak model, the complexity bounds are not very diverse.

*(1)* RCDP($\mathcal{L}_Q$). In contrast to Theorem 4.1, RCDP(CQ) for completable $c$-instances is $\Sigma_3^p$-complete rather than $\Pi_3^p$-complete. Here RCDP(FP) remains undecidable, as opposed its counterpart in the weak model (Theorem 5.1).

**Theorem 6.1:** For completable $c$-instances, RCDP($\mathcal{L}_Q$) is

- undecidable when $\mathcal{L}_Q$ is FO or FP, and
- $\Sigma_3^p$-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$.

The complexity is unchanged when $D_m$ and $V$ are fixed. □

PROOF. (1) For FO and FP, RCDP is already undecidable for ground instances, as shown in the proof of Theorem 4.1. It remains undecidable for $c$-instances since completable ground instances are $c$-instances themselves.

(2) We first show that RCDP(CQ) is $\Sigma_3^p$-hard, by reduction from the $\exists^*\forall^*\exists^*$3SAT problem. We then provide a $\Sigma_3^p$ algorithm for checking whether a $c$-instance is completable for an $\exists$FO$^+$query, by detecting whether there exists a relatively complete ground instance. The algorithm makes use of a characterization of relatively completable $c$-instances.

The lower bound proofs use only fixed $D_m$ and $V$. □

*(2)* RCQP($\mathcal{L}_Q$). In contrast to Theorem 5.1, RCQP($\mathcal{L}_Q$) is no longer trivial for completable $c$-instances when $\mathcal{L}_Q$ is FP. One can verify that the analogy of Lemma 4.2 still holds in this setting. As a result, RCQP for relatively completable $c$-instances coincides with RCQP for ground instances. For the latter, the complexity results are already established by Theorem 4.3. From these the corollary below follows.

**Corollary 6.2:** For completable $c$-instances, RCQP($\mathcal{L}_Q$) is

- undecidable when $\mathcal{L}_Q$ is FO or FP, and
- NEXPTIME-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$.

The complexity is unchanged when $D_m$ and $V$ are fixed. □

*(3)* MinP($\mathcal{L}_Q$). For completable $c$-instances, MinP($\mathcal{L}_Q$) becomes $\Sigma_3^p$-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, rather than $\Pi_3^p$-complete as in the strong model. The complexity bound is rather robust: it is the same for CQ, UCQ and $\exists$FO$^+$, as opposed to their counterparts in the weak model.

**Corollary 6.3:** MinP($\mathcal{L}_Q$) is

- undecidable for completable $c$-instances and for ground instances when $\mathcal{L}_Q$ is FO or FP, and
- $\Sigma_3^p$-complete for $c$-instances and $\Delta_3^p$-complete for ground instances, when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$.

The complexity is unchanged when $D_m$ and $V$ are fixed. □

PROOF. (1) For FO or FP, the proofs of the undecidability of MinP for completable instances are the same as their counterpart of Theorem 4.5 in the strong model.

(2) For $c$-instances, we first show that MinP(CQ) is $\Sigma_3^p$-hard by reduction from $\exists^*\forall^*\exists^*$3SAT. The reduction uses fixed $D_m$ and $V$ only. We then show that MinP($\exists$FO$^+$) is in $\Sigma_3^p$ by giving a $\Sigma_3^p$ algorithm to decide whether a $c$-instance is minimally completable. For ground instances, the notions of completable and strongly complete coincide, and hence, the $\Delta_3^p$-completeness of Theorem 4.5 carries over here. □

*(4)* BdnP($\mathcal{L}_Q$). As remarked above, the analogy of Lemma 4.2 holds on completable $c$-instances. Hence in this setting BdnP for $c$-instances and BdnP for ground instances are equivalent. Moreover, as observed in Section 2.2 the notions of strongly complete and completable coincide on ground instances. For ground instances BdnP has been settled by Theorem 4.6. As a result, we have the following.

**Corollary 6.4:** BdnP($\mathcal{L}_Q$) is

- undecidable for FO and FP, when $K \geq 0$,
- $\Sigma_3^p$-complete for CQ when $K \geq 14$, and for UCQ and $\exists$FO$^+$when $K \geq 1$,

for both completable ground instances and $c$-instances. □

## 7. Conclusions

We have proposed three models to specify the relative information completeness of databases from which both tuples and values may be missing. We have studied the interaction between the analysis of relative completeness and the analysis of data consistency. We have also identified four fundamental problems associated with relative completeness, namely, RCQP, RCDP, MinP and BdnP. For a variety of query languages, we have established upper and lower bounds on these problems, *all matching*, in each of the three completeness models, both for $c$-instances and for ground instances. We expect that these results will help database users decide whether their queries can find complete answers in a database, and moreover, help developers of MDM or databases identify a minimal amount of information to collect in order to answer queries commonly issued.

We summarize the main complexity results in Table 1, annotated with their corresponding theorems. We show the complexity bounds for ground instances (enclosed in parentheses) when they differ from their counterparts for $c$-instances. From the table we can see that different combinations of query languages, completeness models, and the presence and the absence of missing values lead to a spectrum of decision problems with different complexity bounds.

The study of relative information completeness is still in its infancy. An open issue is about the complexity of RCQP for FO in the weak model. We only know that it is undecidable for ground instances, and our conjecture is that it is also undecidable for $c$-instances. (Recall that in the weak model the existence of complete $c$-instances does not imply the existence of complete ground instances.) Another open issue

| $\mathcal{L}_Q$ | RCDP($\mathcal{L}_Q$) | RCQP($\mathcal{L}_Q$) | MinP($\mathcal{L}_Q$) | BdnP($\mathcal{L}_Q$) |
|---|---|---|---|---|
| *Strong model* | Theorem 4.1 | Corollary. 4.3 | Theorem 4.5 | Theorem 4.6 |
| FO, FP | undecidable | undecidable | undecidable | undecidable |
| CQ, UCQ, $\exists$FO$^+$ | $\Pi_3^p$-complete ($\Pi_2^p$-complete [9]) | NEXPTIME-complete [9] | $\Pi_3^p$-complete ($\Delta_3^p$-complete) | $\Sigma_3^p$-complete |
| *Weak model* | Theorem 5.1 | Theorem 5.2 | Theorem 5.3 | Theorem 5.4 |
| FO | undecidable | ? (undecidable) | undecidable | undecidable |
| FP | coNEXPTIME-complete | $O(1)$ | coNEXPTIME-complete | coNEXPTIME-complete |
| UCQ, $\exists$FO$^+$ | $\Pi_3^p$-complete | $O(1)$ | $\Pi_4^p$-complete | $\Sigma_4^p$-complete |
| CQ | $\Pi_3^p$-complete | $O(1)$ | coDP-complete | coDP-complete |
| *Completable model* | Theorem 6.1 | Corollary 6.2 | Corollary 6.3 | Corollary 6.4 |
| FO, FP | undecidable | undecidable | undecidable | undecidable |
| CQ, UCQ, $\exists$FO$^+$ | $\Sigma_3^p$-complete ($\Pi_2^p$-complete [9]) | NEXPTIME-complete [9] | $\Sigma_3^p$-complete ($\Delta_3^p$-complete) | $\Sigma_3^p$-complete |

**Table 1: Complexity results in connection with relative completeness**

concerns whether the complexity bounds remain intact when master data and CCs are fixed. While we have answered the question in positive in most cases, some cases are open, especially for BdnP. A third issue is about the data complexity of RCDP and MinP, in terms of the sizes of databases and master data. A fourth topic is to develop representation systems for relatively complete databases, which we have not addressed. Finally, we want to identify tractable special yet practical cases of the decision problems, consider more expressive CCs, and develop efficient heuristic algorithms for the problems with certain performance guarantees.

**Acknowledgments**. Fan and Geerts are supported in part by EPSRC E029213/1, and acknowledge the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under the FET-Open grant agreement FOX, number FP7-ICT-233599.

## 8. References

[1] S. Abiteboul and O. M. Duschka. Complexity of answering queries using materialized views. In *PODS*, 1998.

[2] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[3] S. Abiteboul, P. C. Kanellakis, and G. Grahne. On the representation and querying of sets of possible worlds. *TCS*, 78(1):158–187, 1991.

[4] M. Arenas, L. E. Bertossi, and J. Chomicki. Consistent query answers in inconsistent databases. In *PODS*, 1999.

[5] M. Arenas, J. Pérez, J. Reutter, and C. Riveros. Composition and inversion of schema mappings. *SIGMOD Record*, 38(3), 2009.

[6] J. Chomicki. Consistent query answering: Five easy pieces. In *ICDT*, 2007.

[7] C. Elkan. Independence of logic database queries and updates. In *PODS*, 1990.

[8] W. Fan. Dependencies revisited for improving data quality. In *PODS*, 2008.

[9] W. Fan and F. Geerts. Relative information completeness. In *PODS*, 2009.

[10] W. Fan and L. Libkin. On XML integrity constraints in the presence of DTDs. In *JACM*, 49(3), 368-406, 2002.

[11] F. Geerts and B. Marnette. Static analysis of schema-mappings ensuring oblivious termination. In *ICDT*, 2010.

[12] G. Gottlob and R. Zicari. Closed world databases opened through null values. In *VLDB*, 1988.

[13] G. Grahne. *The Problem of Incomplete Information in Relational Databases*. Springer, 1991.

[14] T. Imieliński and W. Lipski, Jr. Incomplete information in relational databases. *JACM*, 31(4), 1984.

[15] P. G. Kolaitis. Schema mappings, data exchange, and metadata management. In *PODS*, 2005.

[16] M. W. Krentel. Generalizations of Opt P to the polynomial hierarchy. *Theor. Comput. Sci.*, 97(2):183–198, 1992.

[17] M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, 2002.

[18] A. Y. Levy. Obtaining complete answers from incomplete databases. In *VLDB*, 1996.

[19] A. Y. Levy, I. S. Mumick, Y. Sagiv, and O. Shmueli. Equivalence, query-reachability, and satisfiability in datalog extensions. In *PODS*, 1993.

[20] A. Y. Levy and Y. Sagiv. Queries independent of updates. In *VLDB*, 1993.

[21] D. Loshin. *Master Data Management*. Knowledge Integrity, Inc., 2009.

[22] Microsoft. SQL Server 2008 R2 master data services, 2008. *http://www.microsoft.com/sqlserver/2008/en/us/MDS.aspx*.

[23] D. W. Miller et al. Missing prenatal records at a birth center: A communication problem quantified. In *AMIA Annu Symp Proc.*, 2005.

[24] A. Motro. Integrity = validity + completeness. *TODS*, 14(4), 1989.

[25] D. Olteanu, C. Koch, and L. Antova. World-set decompositions: Expressiveness and efficient algorithms. *TCS*, 403(2-3):265–284, 2008.

[26] C. H. Papadimitriou. *Computational Complexity*. AW, 1994.

[27] J. Radcliffe and A. White. Key issues for master data management. Gartner, 2008.

[28] L. Segoufin and V. Vianu. Views and queries: determinacy and rewriting. In *PODS*, 2005.

[29] M. Spielmann. *Abstract State Machines: Verification Problems and Complexity*. PhD thesis, RWTH Aachen, 2000.

[30] R. van der Meyden. Logical approaches to incomplete information: A survey. In *Logics for Databases and Information Systems*. Kluwer, 1998.

[31] M. Vardi. On the integrity of databases with incomplete information. In *PODS*, 1986.